Universidad Autónoma de Madrid
Escuela politécnica superior

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Trabajo Fin de Grado

INTEGRACIÓN DE UN DISPOSITIVO SMARTGLASS EN UNA PLATAFORMA INTERACTIVA MULTIMODAL

Autor: Isaac Gonzalez Gonzalez
Tutor: José Colás Pasamontes

JULIO 2015

INTEGRACIÓN DE UN DISPOSITIVO SMARTGLASS EN UNA PLATAFORMA INTERACTIVA MULTIMODAL

Autor: Isaac Gonzalez Gonzalez
Tutor: José Colás Pasamontes

HTCLab
Dpto. de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
JULIO 2015

Resumen

Resumen

Este trabajo de fin de grado trata sobre la integración de las gafas de realidad aumentada Epson Moverio Bt-200 en la plataforma multimodal de Vocalia N-Terac, que consta de una serie de módulos, que mediante un script programable, puede ofrecer respuestas multimodales variadas.

Para la realización de este proyecto se hizo una aplicación para Android, (ya que estas smartglass usan Android) cuyo objetivo principal era mantener una comunicación video con el servidor. Esta comunicación se mantiene mediante el protocolo SIP, y para ello, puesto que las librerías que incorpora Android sobre este protocolo son poco potentes, se decidió usar la librería Linphone.

La diferencia de esta herramienta con respecto a otras ya existentes, es la universalidad de este producto, ya que la parte de procesamiento de señal es realizada en el servidor. Es decir, en el resto de plataformas para smartglasses, lo más común es que en las propias gafas, o el dispositivo que proceda, se realicen las tareas como el procesado de los comandos de voz o la creación del entorno de realidad aumentada. La ventaja fundamental de esto es que este producto es apto para una gama más amplia de dispositivos, al no necesitar tanta capacidad de procesamiento y al no tener que hacer un soporte específico para este dispositivo, y por tanto la empresa gana flexibilidad a la hora de elegir el dispositivo que más le convenga.

Esta aplicación está pensada para entornos reales, concretamente para entornos de trabajo donde el operario precise de ayuda o datos técnicos y pueda ser incomodo utilizar las manos para conseguirlos. Algunos ejemplos de esto son las cadenas de montaje, la cirugía, aplicaciones militares, etc.

Palabras Clave

SIP, Realidad Aumentada, Plataforma Multimodal, Android, Aplicaciones Móviles

Abstract

This Bachelor Thesis is about the integration of the augmented reality glasses Epson Moverio Bt-200 into the Vocalia multimodal platform N-Terac, which is made up of a number of modules, which through a programmable script, they can offer diverse multimodal responeses.

For the realization of this project, an Android app was made, (since this glasses wear Android) whose main objetive was to establish a video communication with the server. This communication is maintained through the SIP protocol, and for this, since the Android libraries about this protocol are weak, it was decided to use the Linphone library.

The main difference between this tool with the existing ones, is the universality of this product, since the signal processing part is done by the server. It means, in the rest of the smartglasse's platforms, the most common is that the main glasses, or the pertinent device, perform the tasks like the voice commands proccesing or the creation of the augmented reality enviroment. The main advantage of this, is that this product is suitable for a wide range of devices, since they don't require so many proccesing power and don't need an specific suport for this device, and thus giving flexibility to the company when choosing the more convenient device.

This aplication is designed for real enviroments, specifically for work enviroments where the operator precise of help or technical data and could be unconfortable to use his hands to achive them. Some examples of this are the assembly-lines, surgery, militar aplications, etc.

Key words

SIP, Augmented Reality, Multimodal Platform, Android, Mobile Apps

Agradecimientos

A mi tutor José Colás y al departamento de Vocalia por darme la oportunidad de ser parte de este proyecto.

A mi amigo Carlos y a su infinita paciencia por ayudarme a terminar este trabajo

Y a mi familia y amigos por estar siempre ahí.

Índice general

Índice de Figuras

Índice de Tablas

1 Introducción

1.1. Motivación del proyecto

La principal motivación de este proyecto es la integración de un dispositivo tipo smartglass en una plataforma multimodal con el fin de ofrecer una solución que ayude a la mejora de la productividad en ciertos sectores, especialmente aquellos en los que sea necesario el uso de ambas manos, como talleres o cadenas de montaje.

Dicha integración permitirá llevar hasta el usuario información de naturaleza multimodal (voz, vídeo, texto) así como recoger del entorno del mismo información de la misma naturaleza, transportándola a los sistemas de backoffice de la compañía (SAP, Dominó, etc...). Todo ello será posible gracias a la incorporación en las smartglass de elementos como una webcam, un micrófono, sensores de movimiento y principalmente un display translucido que permite al usuario visualizar información sin perder visibilidad.

La realidad aumentada tiene un gran uso potencial en las áreas industriales, incluidas algunas especializaciones médicas como cirugía, donde necesitan más que sus propias manos para asistir el proceso de trabajo.

1.2. Objetivos y enfoque

Dado que esta aplicación está pensada para un uso de la misma en tiempo real, y con el menor uso del táctil posible por parte del usuario, los objetivos a destacar son:

- El desarrollo de una solución software que pueda ejecutarse en la unidad de proceso que tienen las smartglass en el momento actual (la mayor parte están basadas en arquitecturas hardware de smartphones a los que se les ha quitado la parte de telefonía 3g y se ha incorporado tarjetas gráficas preparadas para llevar una imagen, que puede ser estereoscópica, a los displays de la propia smartglass).
- Diseñarla de tal modo que permita una comunicación ininterrumpida con la plataforma mediante audio y vídeo en ambos sentidos.

- Desarrollarla para que sea para el usuario lo más intuitiva posible. Para ello se usarán instrucciones reconocibles como comandos de voz.
- Hacer esta aplicación compatible con las gafas de realidad aumentada Epson Moverio Bt-200. Dado que este dispositivo lleva un sistema operativo Android, la aplicación será desarrollada en este lenguaje.

1.3. Metodología y plan de trabajo

Para este proyecto, debido a la naturaleza evolutiva de esta aplicación, se ha usado una planificación con un ciclo de vida iterativo en el cual se ha ido mejorando el software y añadiendo más funcionalidades en cada iteración.

El objetivo primero era conseguir la comunicación SIP con el servidor en un entorno de desarrollo Android, empezando primero con comunicación voz simple, una vez dominada esta se incluyó la comunicación por vídeo. Después, se han ido añadiendo otros requisitos al proyecto, hasta conseguir la aplicación final. Además esta aplicación sigue en constante crecimiento y se espera añadir en un futuro nuevas funcionalidades.

1.4. Contenido del documento

En el primer capítulo introductorio se trata de mostrar porqué se realizó este proyecto, y un poco por encima que método de desarrollo de software se pensó que era la ideal.

El segundo capítulo, del estado del arte, tratará de poner en situación al lector sobre en que punto se encuentra actualmente esta tecnología, para tratar de averiguar que tipo de aplicación sería conveniente desarrollar y se tratará de justificar por qué se ha elegido el dispositivo que finalmente se usó.

En el tercer capítulo se explica desde un punto de vista modular el diseño planteado para este proyecto. Además se explicará en que protocolos se basa está aplicación y se profundizará en su sintaxis con ejemplos ilustrados.

El cuarto capítulo trata del desarrollo de la aplicación desde un punto de vista más técnico. Se introducen también unos pocos conceptos de Android para mostrar la complejidad del mismo.

En el quinto capítulo se muestran las pruebas de conexión con la plataforma realizadas.

El sexto capítulo habla simplemente sobre las conclusiones y en que dirección se espera que se encamine este proyecto.

2

Estado del arte

2.1. Introducción

En este proyecto se trata de integrar unas smartglass de realidad aumentada en una plataforma especializada para esto. El estado del arte de este proyecto, por tanto, tiene tres partes claramente diferenciadas, gafas de realidad aumentada, plataformas multimodales y realidad aumentada.

2.2. Gafas de Realidad Aumentada.

La realidad aumentada consiste en plasmar objetos o imágenes virtuales sobre el entorno real, para que a los ojos del usuario, este perciba una realidad mixta, con elementos de ambos entornos conectados. En estos últimos años, las gafas de realidad aumentada están mejorando considerablemente sus prestaciones, algunos de los modelos más representativos de este sector son:

- DAQRI: DAQRI no es técnicamente una gafa, es en realidad un casco pensado especialmente para entornos industriales y para que sirva también de protección, posee cámaras que permiten visión 360 grados, visión estereoscópica y SO Android. Este dispositivo aún no está disponible en el mercado, por lo que no hay mucha más información sobre este, pero ya se está empezando a hacer oír y todo parece apuntar a que será un referente en este campo. Se sabe que su precio superará los 1500$. Además, esta empresa también comercializa su propia herramienta para la creación de aplicaciones de realidad aumentada.
- Google Glass: Las controvertidas gafas de Google llevan una cámara capaz de tomar fotos a 5 megapíxeles y grabar video a 720p, no poseen visión estereoscópica ya que solo incorpora un display, están pensadas para un uso cotidiano, obviamente llevan SO Android e incorporan numerosos sensores que son: giroscopio de 3 ejes, acelerómetro de 3 ejes, sensor geomagnético (brújula), sensor de luz ambiente y sensor de proximidad interno. Su precio está en torno a 1000€.
- Microsoft Hololens: Estas gafas parece que serán un gran referente en el campo de la realidad aumentada en los próximos años. Aún no están a la venta, y por tanto, como

suele pasar, la información que se da a conocer sobre este dispositivo es más bien escasa, pero se sabe que incorporarán Windows10, visión estereoscópica, y se intuye que tendrán sensación inmersiva, es decir, que no se verá una pantalla, si no que los objetos se plasmarán sobre la misma visión del usuario.

- Epson Moverio BT-200: Finalmente estas son las gafas utilizadas en el proyecto. Poseen visión estereoscópica, cámara VGA, y su precio es de unos 700€. Vienen además con un panel táctil, (similar a un smartphone, pero sin lcd) para su manejo, además incorporan sensores como son: GPS, brújula, acelerómetro y giroscopio.

Dispositivos	LCDs	Cámara	RAM	ROM	SO	Precio	Disp.
DAQRI	2	360 grados	Desc.	Desc.	Android	$>$ 1500$	No
Google Glass	1x(640x360)	5 MP 720p	2GB	16GB	Android	1000€	No
Microsoft Hololens	Desc.	Desc.	2GB	Desc.	Windows	Ĩ000$	No
Epson Moverio BT-200	2x(960x540)	VGA	1GB	8GB	Android	700€	Sí

Cuadro 2.1: Información de los distintos dispositivos

Como se puede ver, la mayoría estas gafas apuestan por usar como SO Android, ya que es un sistema muy extendido, lo que hace que sea más fácil encontrar desarrolladores para el mismo.

Por supuesto existen infinidad de gafas en el mercado, pero se han escogido estas para hablar de ellas porque o bien son muy representativas, o bien están pensadas para entornos industriales, como es el caso de este proyecto.

2.3. Plataformas interactivas para SmartGlass

Existen también diversas plataformas interactivas para SmartGlass, aunque no son demasiadas, vale la pena mencionarlas.

La mayoría de ellas tienen su software propio, que se instala en el dispositivo y se conecta con la plataforma mediante los protocolos pertinentes, para básicamente tener almacenado en la nube los patrones a reconocer, tanto de imagen como de audio. Alguna de estas son: Skylight, una plataforma interactiva de APX Labs, Atheer Labs, Augmented Ltd y N-Terac.

Esta última, será la que usemos para este proyecto para la integración de las smartglass. Se describirá en los puntos siguientes.

2.4. 2.3 SDKs de realidad aumentada

Para desarrollar aplicaciones de realidad aumentada, existen distintos SDKs que permiten crear estas aplicaciones de un modo algo más sencillo, simplificando mucho la tarea. Existen muchos SDKs, no obstante, para entornos Android, los más importante son:

- Vuforia: Este SDK está producido por Qualcomm, que también se dedica al desarrollo de procesadores para móviles. Aquí los objetos se insertan como vectores de puntos en

archivos .h, lo que a priori lo hace algo más complicado que metaio por ejemplo, en el que los objetos son directamente archivos 3D. El fuerte de este SDK es claramente su robusto tracking de imágenes y que su versión gratuita es muy completa con respecto a la de pago. Como parte negativa la documentación no es demasiado clara, y su uso es algo complejo.

- Metaio: Metaio ha sido recientemente adquirido por Apple, es un SDK muy robusto. Su principal ventaja es su facilidad de uso, de tal manera es relativamente sencillo conseguir buenos resultados comparado con Vuforia por ejemplo. Por el contrario, su sistema de tracking no es tan preciso como otros, aunque es altamente parametrizable.
- Wikitude: Inicialmente estaba ideado para realidad aumentada basada en localización, aunque desde 2012 incluye también tracking de imágenes. Al igual que metaio, es posible su uso gratuito con una marca de agua, salvo por unas pocas características como reconocimiento en la nube.

SDK	3D Object Traking	Natural Feature	GPS	IMU Sensor	Marker	Visual Search	Face Tracking
Vuforia	Sólo caja o cilindro	Sí	No	Sí	Sí	Sí	No
Wikitude	Sí	Sí	Sí	Sí	Sí	Basado en la nube	No
Metaio	Sí	Sí	Sí	Sí	Sí	Sí	Sí

Cuadro 2.2: Información de los distintos SDKs

3 Diseño

3.1. Punto de partida

El objetivo de este proyecto es el desarrollo de una interfaz de usuario que se ejecute en el terminal (smartglass) y que se conecte con la plataforma para enviar y recibir información. Para ello contamos con la plataforma de Vocalia N-Terac y con el terminal finalmente elegido que serán las Epson Moverio Bt-200.

3.1.1. Descripción de la plataforma

Lo que diferencia a la plataforma N-Terac de otras aplicaciones multimodales, es el carácter universal de esta, es decir, casi cualquier dispositivo es apto para conectarse con esta plataforma sin tener que desarrollar (en la mayor parte de los casos) un software específico para este.

Esto es posible porque el procesado de la información, como el tracking y el incrustado de los objetos de realidad aumentada en la imagen, se harán en la plataforma, en vez de en el terminal, haciendo que la carga en el dispositivo sea mínima.

Para ello el dispositivo se comunica mediante audio y vídeo con la plataforma a través del protocolo SIP, y recibe la información extra (texto o imágenes), si la necesitase mediante peticiones HTTP.

3.1.2. Diagrama de bloques

En la figura 3.1 se muestra el diagrama de bloques de la plataforma N-Terac. Un cliente se conectaría al firewall router y a través de este puede interactuar con el web server para servicios generales o a un agente virtual, si este cliente tiene uno asignado.

El agente virtual es un nodo de servicio que tiene varios módulos disponibles con los que el cliente puede interactuar a través del dialog control de cada virtual agent. A continuación se detallará que hace cada módulo.

- **Hub server client:** es un servidor TCP al que se conectan todos los módulos cliente que

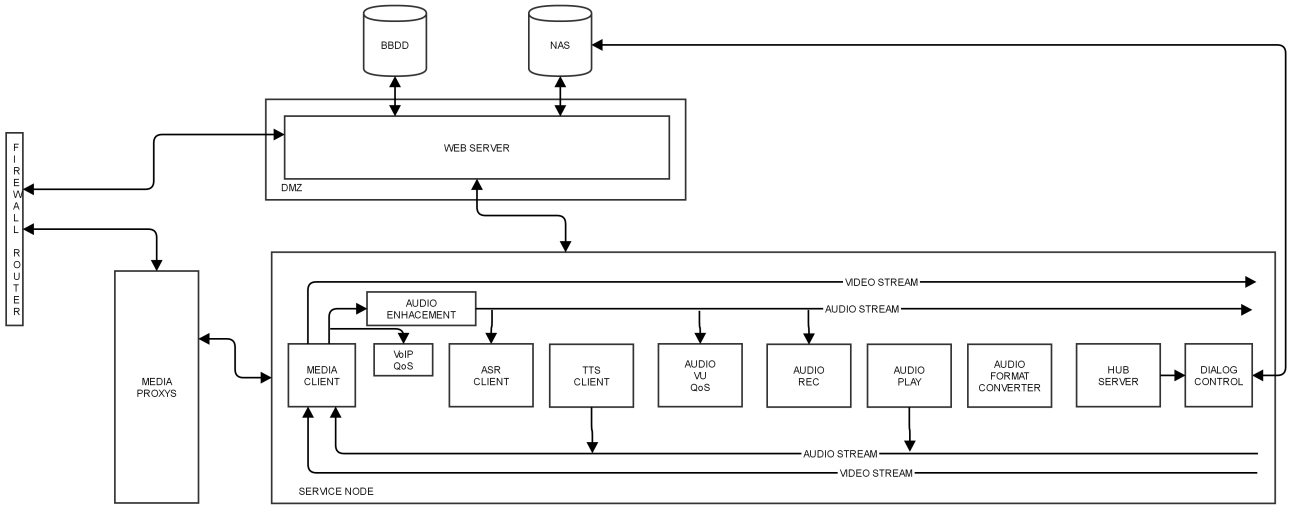

Figura 3.1: Diagrama de Módulos N-Terac

están arrancados en un nodo de servicio. Su función es la redistribución de mensajes entre módulo de dialogo y el resto de módulos.

- **Media client:** es el softphone de la plataforma. El cliente obtiene su dirección y se conecta a él a través de SIP. Este no inserta directamente el vídeo, pero es el encargado de transmitir el audio y el video al cliente desde la plataforma. La tarea de insertar el vídeo recae en el Video Player client, que lo insertará en el video stream cuando sea ordenado por el diálogo.
- **VoIP QoS client:** Mide la calidad del audio en base a los paquetes perdidos.
- **ASR client:** Módulo cliente TCP con funcionalidad de reconocimiento automático de habla independiente del locutor y gran vocabulario. Basado en HMM (Modelos Ocultos de Markov) a nivel fonético, permite el desarrollo de servicios o aplicaciones con capacidad de interacción vocal con el usuario.
- **TTS client:** Módulo de Síntesis de Habla a partir de texto (Text-to-Speech) capaz de convertir un texto en una señal de audio que es enviada al Media Client (VoIP soft phone).
- **Audio VU-QoS client:** este cliente implementa un VU de audio para medir en tiempo real el nivel de señal recibido y detectar situaciones de muy bajo nivel o de muy alto nivel (saturación) que pueden conducir a una degradación del reconocimiento de habla o a ficheros grabados con muy poca calidad. Este módulo, basándose en un análisis espectral, mide la calidad del audio de entrada, detectando situaciones de mucho ruido (baja SNR). Ambas informaciones son enviadas a la base de datos (Servicios Generales) de forma periódica cada 500 ms o 1 s, con el fin de que las interfaces de usuario puedan leerlas e informar al usuario de situaciones conflictivas (demasiada energía de señal o muy baja-ajustar volumen de la tarjeta de sonido-y bajas SNR-evitar ese entorno ruidoso si es posible).
- **AudioRec client:** Cuando recibe una orden del dialog, se pone a grabar el stream de entrada de audio y lo guarda en un archivo .wma.
- **AudioPlayer client:** Cuando sea necesario una respuesta por voz, este bloque accionado por el dialog cargará un archivo .wma en el stream de audio y este llegará al cliente a través del media client.

- **Audio Format converter:** Bloque capaz de convertir distintos tipos de formatos de audio al formato .wma, que es el que usa la plataforma
- **Dialog control:** Generador de órdenes. Ejecuta una lógica funcional definida en un script. Es el encargado de activar el resto de bloques por medio de mensajes.

3.2. Smartglass a Integrar

Como ya hemos comentado, las smartglass escogidas para este proyecto fueron las Epson Moverio Bt-200, ver figura 3.2, principalmente por su precio y por su disponibilidad inmediata. Aunque estas smartglass no son muy punteras en cuanto a prestaciones, cumple de sobra con los requisitos.

Figura 3.2: Gafas Epson Moverio

A continuación, en la tabla 3.1 se muestran algunas de las principales características del dispositivo.

Aspecto del dispositivo	
Método conductor LCD	Matriz activa TFT de polisilicio
Tamaño LCD	0,42 pulgada panel ancho (16:9)
Número de píxeles LCD	518.400 Puntos (960x540) x 3
Campo de visión	aproximadamente 23 grados
Tamaño de pantalla (distancia de proyección)	40 pulgadas en 2,5 m - 320 pulgadas en 20 m
Reproducción del color	24 bits color (16,77 millones de colores)
Frecuencia de actualización	60 Hz
Tipo de SO	Android 4.0.4
Sensores del dispositivo	
Cámara	VGA
GPS	Sí, en el controlador
Brújula	Sí
Giroscopio	Sí
Acelerómetro	Sí

Micrófono	Sí, tanto en los auriculares como en el controlador
Conectividad	
LAN inalámbrica	IEEE 802.11b/g/n
Bluetooth	3
microUSB	USB 2.0
CPU y Memoria	
CPU	TI OMAP 4460 1.2GHz Dual Core
RAM	1 GB
Memoria interna	8 GB
Memoria externa	microSD (máximo 2 GB), microSDHC (máximo 32 GB)

Cuadro 3.1: CPU y Memoria del dispositivo

3.3. Solución: Integración de la smartglass en N-Terac

El diagrama 3.3 muestra, por bloques, la comunicación que se establece entre las smartglass y la plataforma.

Primeramente el terminal se registra en el web server, y este le manda los parámetros de conexión, que son los datos para registrarse en el SIP y la dirección del nodo de servicio asignado al usuario.

Después se registra en el media proxy con los datos recibidos y solicita la ejecución del script de diálogo (funcionalidad del servicio). Una vez activado, el ASR (Automatic Speech Recognition) reconoce los patrones de voz y este manda comandos al dialog, que se encarga de poner a disposición del usuario el recurso o realizar la acción solicitada.

En caso de que el usuario solicitase un recurso de vídeo, simplemente el dialog mandaría una orden al player video con que vídeo cargar en el stream de vídeo y una vez cargado el phone ID asociado lo haría llegar hasta las smarglass.

También es posible solicitar al dailog grabar el vídeo para que se guarde en el servidor. Si el usuario solicita un texto o una imagen, el dialog asignará la URL de ese recurso a una base de datos, y el usuario accederá a este a través del web server.

3.3.1. App Android

El objetivo es el desarrollo de una interfaz de usuario que se ejecute en el terminal (smartglass) para conectar con la plataforma y enviar/recibir información.

Esta es una aplicación poder integrar un dispositivo en la plataforma con una funcionalidad de comunicación multimodal, con transferencia audio y vídeo en ambos sentidos y por otro lado la interacción (control) con la plataforma y los servicios que en ella se ejecutan.

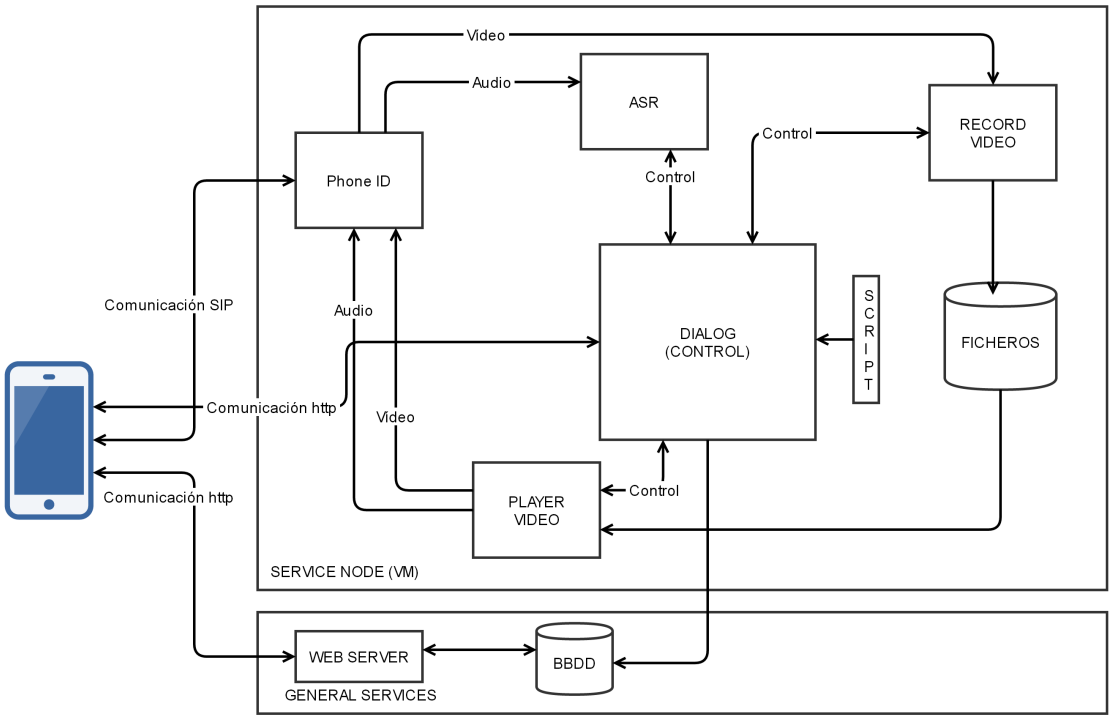

Figura 3.3: Diagrama de Interacción entre las smartglass y la plataforma

3.3.2. Interacción con la plataforma

Protocolos a utilizar

Los protocolos que se utilizan en esta aplicación son el protocolo SIP para las comunicaciones de audio y vídeo y el protocolo HTTP para obtener los parámetros de conexión y recursos, si los hubiese, de otra naturaleza, ya sea imagen, texto...

El protocolo SIP actual es un estándar para las comunicaciones multimedia por internet, muy parecido a HTTP en cuanto a que son mensajes ASCII. Se trata de un protocolo ligero capaz de establecer llamadas entre el que llama y el que es llamado a través de una red IP. Además es capaz de determinar la dirección IP actual del llamado, siempre que este esté registrado en el servidor SIP, y además proporciona mecanismos para la gestión de llamadas tales como añadir nuevos flujos multimedia durante la llamada, cambiar el método de codificación o invitar a nuevos participantes durante la llamada, además de mecanismos para la transferencia de llamadas y el mantenimiento de estas. [1]

El ejemplo de la figura 3.4 es un caso similar al que nos ocupa, ya que también en este caso las llamadas son redirigidas a través del proxy. Previamente a que Alice pueda llamar a Boris, al menos Boris ha tenido que mandar un REGISTER al SIP proxy, para que este pueda localizar la dirección en la que se encuentra Boris actualmente. Una vez Boris se ha registrado, cuando Alice le mande un INVITE a su proxy con la dirección SIP de Boris, (las direcciones SIP son similares a direcciones de correo electrónico) el proxy será capaz de redireccionar ese mensaje al usuario Boris, ya que el proxy habrá guardado la dirección IP actual de Boris. Una vez el mensaje le llegue a Boris, este responderá con un OK (en el caso de que Boris quiera hablar con Alice) y ambos teléfonos SIP automáticamente negociarán los parámetros de la llamada, según una lista de códecs que cada teléfono tendrá asignada con una prioridad distinta.

Una vez establecidos los parámetros de la llamada (o videollamada en este caso), Alice podrá

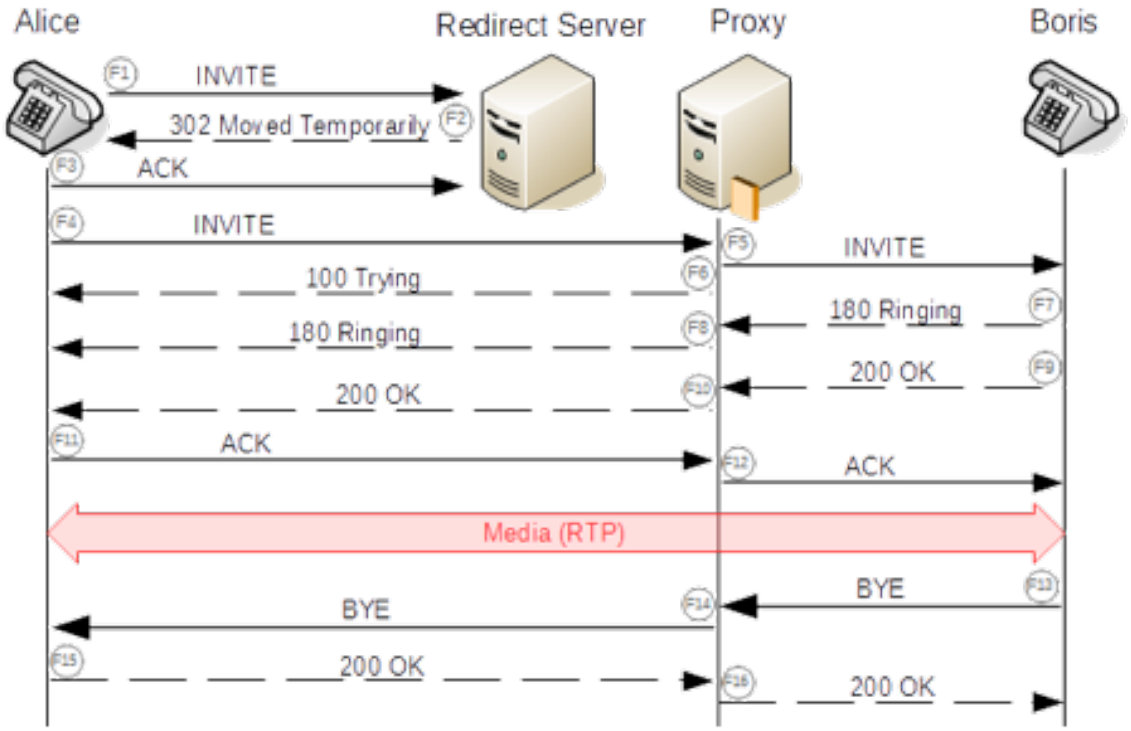

Figura 3.4: Diagrama de Comunicación SIP entre las smartglass y la plataforma

hablar con Boris a través de RTP (Real-time Tranport Protocol), un protocolo utilizado para la transmisión multimedia (generalmente audio y vídeo) en tiempo real, pasando también este flujo a través del proxy.

Esta aplicación también usa protocolo HTTP para la comunicación con el web server. El protocolo HTTP es el protocolo de la capa de aplicación de la web. HTTP se implementa en dos programas, cliente y servidor. El programa cliente y el servidor, que se ejecutan en sistemas terminales diferentes, se comunican entre sí intercambiando mensajes HTTP. HTTP define la estructura de estos mensajes y como el cliente y el servidor intercambian los mensajes. [1]

Los mensajes HTTP son mensajes simples que van sobre TCP y usualmente usan el puerto 80. Un ejemplo de una petición HTTP se muestra en la figura 3.5

```
GET /index.html HTTP/1.1
Host: www.example.com
User-Agent: nombre-cliente
[Línea en blanco]
```

Figura 3.5: Ejemplo de Petición HTTP

El tipo de petición por parte del cliente puede ser de varios tipos, pero las más conocidas sin duda son las peticiones de tipo GET y las peticiones de tipo POST. Las peticiones de tipo GET sirven para solicitar datos de un recurso específico. En las aplicaciones en las que la seguridad sea un parámetro crítico se debería evitar usar este tipo de petición, ya que los recursos se envían dentro de la URI. Las peticiones tipo POST, envían datos para que sean procesadas pero es más seguro que GET ya que estos parámetros están en la URI.

La respuesta entonces del servidor iría precedida de un código decimal de tres cifras que indica la validez de la respuesta, en la tabla 3.2 se pueden ver los distintos tipos.

1xx	Mensajes
2xx	Operación exitosa
3xx	Redirección
4xx	Error por parte del cliente
5xx	Error por parte del servidor

Cuadro 3.2: Tipos de respuesta HTTP

Flujos de Mensajes y Datos

La aplicación manda, nada más iniciarse, una petición HTTP para obtener los parámetros de registro, que son leídos como un JSON. Estos parámetros son: el nombre de usuario, la contraseña, el servidor y la dirección SIP del nodo de servicio, que incorpora un softphone de audio y vídeo.

Una vez obtenidos estos parámetros, los usa para registrarse en la plataforma mediante un SIP REGISTER clásico.

Una vez registrado, cuando el usuario le dé al botón "videocall" comienza una videollamada por SIP y a su vez, paralelamente, la aplicación comenzará a realizar peticiones HTTP a un archivo PHP del servidor, en el que leerá en un JSON si hay algún recurso que mostrar, y en caso de que lo hubiese, la dirección de este. El diagrama 3.6 ilustra este comportamiento.

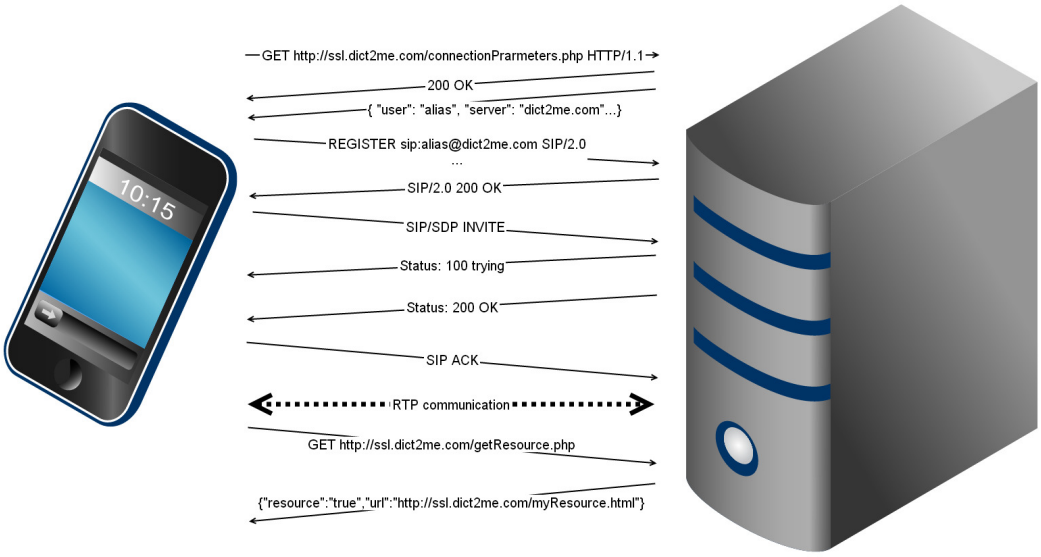

Figura 3.6: Flujo de Datos entre el Terminal y el Servidor

4

Desarrollo

En esta sección se tratará de explicar el desarrollo de la aplicación creada para integrar las smartglass en la plataforma. Esta aplicación ha sido creada en Android, ya que como se ha comentado antes, es el lenguaje en el que están desarrolladas las smartglass.

4.1. Conceptos sobre Android

Antes de comenzar a explicar el desarrollo de la aplicación, conviene empezar explicando una serie de conceptos en Android para un más fácil entendimiento del desarrollo de la misma. Android es un sistema operativo que combina el lenguaje de java para la lógica de la aplicación con XML que sirve tanto para la interfaz de usuario como para el AndroidManifest. En los dispositivos Android, los programas se ejecutan en una máquina virtual llamada Dalvik, esto hace que las aplicaciones tarden en ser lanzadas respecto a si estuviesen en código nativo.

4.1.1. Activities y su lifecycle

Una Activity es un componente de la aplicación que provee una pantalla con la cual los usuarios pueden interactuar para hacer algo, como marcar un número, tomar una foto, enviar un email o ver un mapa. A cada activity se le da una ventana en la cual dibujar su interfaz de usuario. [10]

Las activities, al iniciarse siguen un ciclo de vida o lifecycle, según se muestra en la figura 4.1. Una activity, nada más iniciarse, llamará a su método onCreate(), seguidamente a onStart() y así hasta que sea destruida y llame a su método onDestroy().

Todas las activities, si no se especifica otra cosa, se ejecutan en el mismo hilo, el hilo principal. Es por ello que si la activity tuviese un bucle largo de resolver en el tiempo y no fuese capaz de responder a un evento de la interfaz de usuario rápidamente (5 segundos) la aplicación mostraría automáticamente el fatídico mensaje de la figura 4.2.

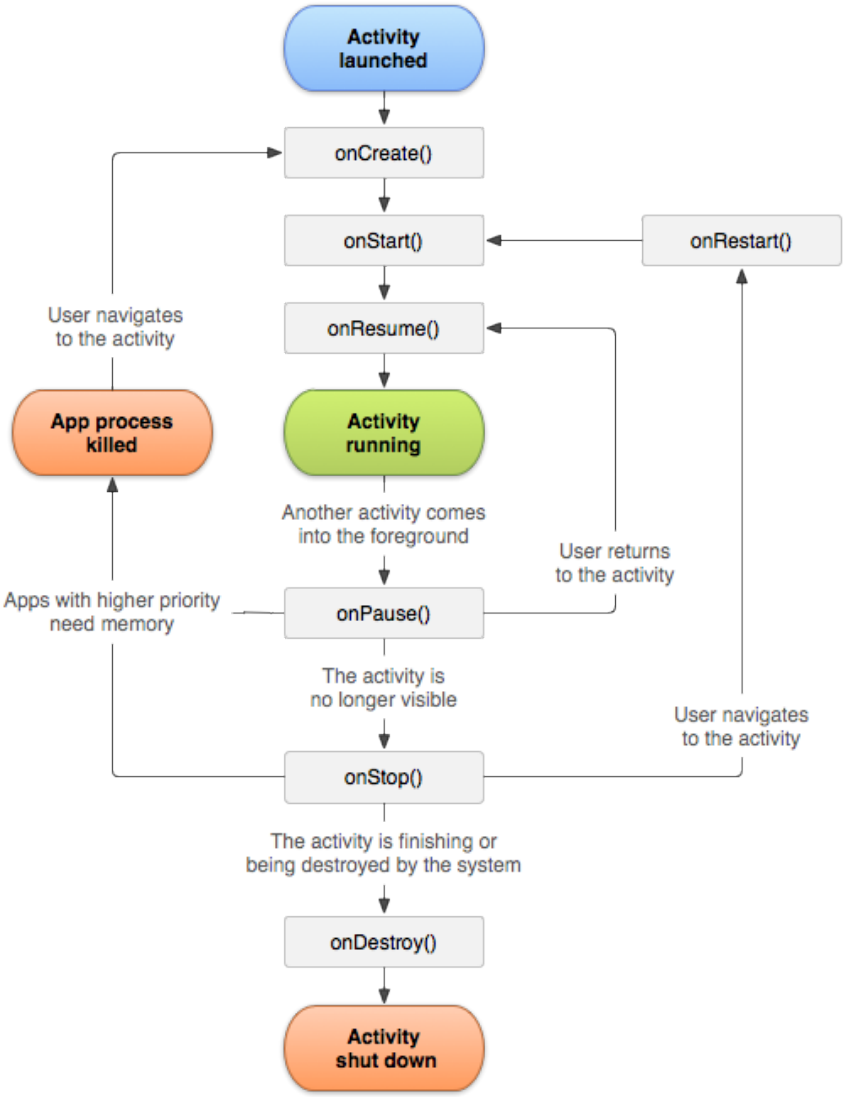

Figura 4.1: Lifecycle de una Activity

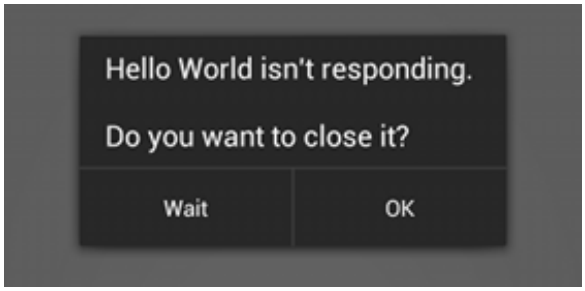

Figura 4.2: Ejemplo de error

4.1.2. Services

Un service en Android, es similar a una activity solo que sin el elemento de la interfaz de usuario. Los services tienen un lifecycle distinto, como indica la figura 4.3, dependiendo de si se inician llamando al método bindService(), por el que el cliente (la activity) queda unido al servicio y monitorizará su conexión mediante la clase ServiceConnection implementada dentro de la propia activity (en nuestro caso), o si es creado mediante el método startService():

Esto es útil en ciertas ocasiones, pero puede traer algún quebradero de cabeza, ya que al ejecutarse en el mismo hilo que la activity que haya en ese momento (por defecto), tampoco puede tener operaciones que ocupen más de 5 segundos en el tiempo, ya que dejaría bloqueada la actividad y esta no podría responder a los eventos de la IU.

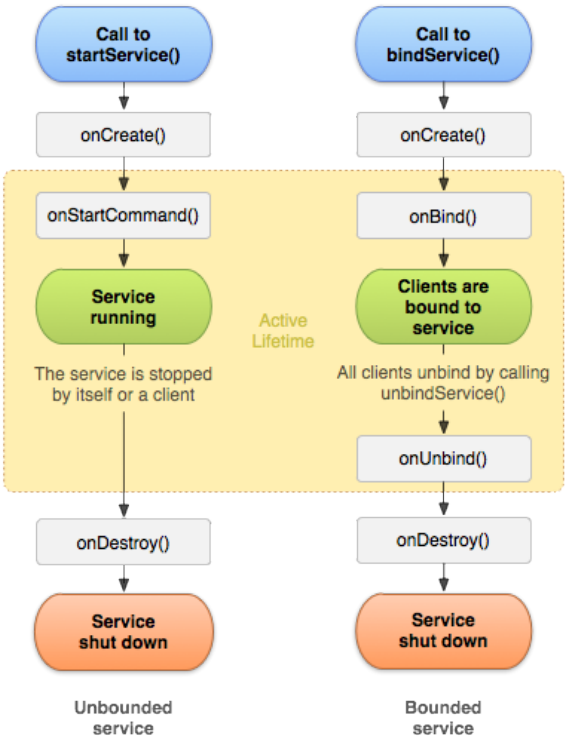

Figura 4.3: Lifecycle de un servicio

4.1.3. Threads

Para evitar estos cuelgues existen los Threads. Cuando se define un nuevo thread, la aplicación ejecuta el Runnable de ese thread en un hilo diferente, permitiendo así que el sistema pueda ejecutar varias tareas "a la vez".

4.2. La aplicación

La aplicación nada más iniciarse, desde su MainActivity lee en un fichero interno de la aplicación una URL de la que obtendrá los parámetros de registro y un service node con el que comunicarse que esté disponible, en forma de dirección SIP. Estos parámetros son leídos por la aplicación lanzando previamente un hilo distinto que realizará una petición HTTP a la URL antes mencionada, y recibirá estos parámetros como elementos JSON. Una vez obtenidos, este

hilo morirá y se creará otro que se encargará de registrarse en el SIP proxy. Además de esto, este hilo tendrá asociado un servicio que previamente fue arrancado por la MainActivity, que será el encargado de recibir notificaciones (listener) ante posibles eventos SIP y notificar mediante un Broadcast al resto de activities.

Una vez que la aplicación está registrada en el servidor, cuando el usuario pinche en un botón para hacer la videollamada, la aplicación cambiará a otra activity con simplemente dos elementos (a simple vista) en su IU. Una GLSurfaceView, que será el elemento encargado de reproducir el vídeo que le llegue a la aplicación desde el softphone y un botón para finalizar la llamada. Por detrás, esta activity estará gestionando la videollamada para mandar el contenido de la propia webcam y plasmar el recibido en el GLSurfaceView de la IU.

Además de esto, esta activity nada más empezar, lo que hará será lanzar un nuevo hilo que preguntará periódicamente al webserver si hay algún recurso para mostrar. En el lado del servidor, si el ASR del service node reconoce que el usuario ha pedido algún recurso, informará al dialog, que a su vez informará al webserver para que envíe al usuario la URL del recurso como respuesta JSON en la url en la que el usuario está preguntando periódicamente.

4.2.1. Librerías utilizadas

La única librería externa utilizada para este proyecto ha sido la librería Linphone, que proporciona las herramientas necesarias para gestionar las comunicaciones SIP en Android. Se miraron también otras librerías SIP para Android, como pjsip pero se descartó porque en el momento de empezar con el proyecto no tenía videollamadas para Android.

La utilización de esta librería es un tanto particular. En primer lugar, se necesita tener instanciado un objeto LinphoneCore y otro objeto LinphoneCoreFactory para la gestión de las comunicaciones. Para solucionar el hecho de que solo pueda haber una instancia, se ha creado una clase llamada MyLinphone.class que guarda como variable estática cada uno de estos objetos, de tal modo, que cuando una clase necesita de alguno de estos objetos, sacará la instancia de esta clase. Además, cuando se quiere hacer una videollamada, es obligatorio plasmar el vídeo entrante en un GLSurfaceView y el vídeo saliente (la webcam) en un SurfaceView.

4.2.2. Diagrama temporal de hilos simplificado

Siendo cada columna un hilo, en el diagrama de la figura 4.4 se trata de mostrar la iteración entre los hilos de la aplicación y las actividades. El thread JSON es iniciado por la MainActivity y en cuanto obtiene los parámetros JSON, la MainActivity recoge los parámetros (aunque no directamente desde hilo). Una vez terminada esta tarea el hilo muere. También la MainActivity activará el hilo Linphone que se encargará de recibir todos los eventos SIP (como registro en el servidor correcto, llamada entrante, fin de llamada, nuevo mensaje, etc.). Este hilo estará activo mientras dure la aplicación.

Una vez la videollamada ha comenzado, VideocallActivity creará el hilo Webserver, que será el encargado de realizar las peticiones HTTP a la plataforma para recibir los recursos que fueran necesarios.

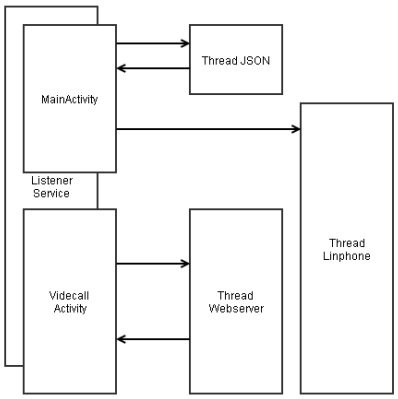

Figura 4.4: Diagrama temporal de los hilos de la aplicación

5

Integración, pruebas y resultados

5.1. Integración de las smartglass en la plataforma

Con el objetivo de comprobar la funcionalidad de la aplicación Android, se conectaron las smarglass a la plataforma N-Terac a través de una videollamada.

Esta prueba supone primero la solicitud de un nodo de servicio a la plataforma con el correspondiente envío de información por parte de la plataforma, el registro del teléfono en el SIP proxy y a continuación la videollamada con el media client para comprobar su funcionalidad.

Figura 5.1: Versión de escritorio de la plataforma

En la figura 5.1 se pueden apreciar los módulos descritos en anteriores capítulos como cada una de las cajas que aparecen.

Se puede ver en el WebPhone la transmisión de vídeo recibida desde el terminal y el resto de módulos activos. Sólo están activos los que desde el dialog hayan sido activados.

No.	Time	Source	Destination	Protocol	Length	Info
408	7.416139	192.168.1.102	213.149.230.100	SIP	568	Request: REGISTER sip:ssl.dict2me.com:5061 (1 binding) |
413	7.539934	213.149.230.100	192.168.1.102	SIP	503	Status: 200 OK (1 binding) |
420	7.593717	192.168.1.102	213.149.230.100	SIP	568	Request: REGISTER sip:ssl.dict2me.com:5061 (1 binding) |
425	7.716971	213.149.230.100	192.168.1.102	SIP	504	Status: 200 OK (1 binding) |
624	11.879647	192.168.1.102	213.149.230.100	SIP/SDP	1172	Request: INVITE sip:alberto@ssl.dict2me.com |
625	11.985372	213.149.230.100	192.168.1.102	SIP	365	Status: 100 Trying |
626	11.985591	213.149.230.100	192.168.1.102	SIP	587	Status: 180 Ringing |
628	12.165249	213.149.230.100	192.168.1.102	SIP/SDP	815	Status: 200 OK |
629	12.194742	192.168.1.102	213.149.230.100	SIP	324	Request: ACK sip:alberto@213.149.230.100:5061 |
632	12.410046	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=0, Time=4240
633	12.420538	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=1, Time=4400
634	12.440175	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=2, Time=4560
635	12.469890	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=3, Time=4720
636	12.480277	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=719, Time=52073
637	12.480452	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=719, Time=52073
638	12.480477	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=720, Time=52233
639	12.480512	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=720, Time=52233
640	12.480527	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=721, Time=52393
641	12.480591	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=4, Time=4880
642	12.481539	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=721, Time=52393
643	12.500180	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=5, Time=5040
644	12.520099	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=6, Time=5200
645	12.539915	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=7, Time=5360
646	12.560598	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=8, Time=5520
647	12.576153	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=722, Time=52553
648	12.576523	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=722, Time=52553
649	12.576565	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=723, Time=52713
650	12.576596	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=723, Time=52713
651	12.576660	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=724, Time=52873
652	12.577567	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=724, Time=52873
653	12.577608	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=725, Time=53033

Figura 5.2: Invite del terminal

En la figura 5.2 se muestra primero el registro del terminal en la plataforma, como se ha comentado antes, mediante un SIP REGISTER, y acto seguido una invitación estándar a una conferencia también desde el terminal a la plataforma.

Una vez la llamada ha sido aceptada, comienza el intercambio de tramas mediante protocolo RTP.

No.	Time	Source	Destination	Protocol	Length	Info
3572	31.220908	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=940, Time=154640
3573	31.236032	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1659, Time=202473
3574	31.237549	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1659, Time=202473
3575	31.240855	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=941, Time=154800
3576	31.253688	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1660, Time=202633
3577	31.254249	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1660, Time=202633
3578	31.260655	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1661, Time=202793
3579	31.262174	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1661, Time=202793
3580	31.263801	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=942, Time=154960
3581	31.280912	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=943, Time=155120
3582	31.291497	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1662, Time=202953
3583	31.293086	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1662, Time=202953
3584	31.300915	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=944, Time=155280
3585	31.306134	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1663, Time=203113
3586	31.307439	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1663, Time=203113
3587	31.320414	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=945, Time=155440
3588	31.323433	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1664, Time=203273
3589	31.324643	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1664, Time=203273
3590	31.340553	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=946, Time=155600
3591	31.354223	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1665, Time=203433
3592	31.355829	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1665, Time=203433
3593	31.360419	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=947, Time=155760
3594	31.371191	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1666, Time=203593
3595	31.371653	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1666, Time=203593
3596	31.380203	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=948, Time=155920
3597	31.385269	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1667, Time=203753
3598	31.386664	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1667, Time=203753
3599	31.399937	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=949, Time=156080
3600	31.418332	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1668, Time=203913
3601	31.418550	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1668, Time=203913
3602	31.419918	192.168.1.102	213.149.230.100	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE8ECDE2E, Seq=950, Time=156240
3603	31.431551	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1669, Time=204073
3604	31.431746	213.149.230.100	192.168.1.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE062, Seq=1669, Time=204073
3605	31.437827	192.168.1.102	10.100.36.4	SIP	379	Request: BYE sip:alberto@10.100.36.4:5061 |

Figura 5.3: BYE del terminal

Después en la figura 5.3 se muestra como el terminal se desconecta de la plataforma enviando un BYE.

6

Conclusiones y trabajo futuro

6.1. Conclusiones

En los últimos años las tecnologías de realidad aumentada están tomando lugar en tareas cotidianas tanto industriales como domésticas a través de dispositivos como las smarglass o los smartphones, cambiando el proceso de realización de diversas tareas añadiendo una nueva dimensión que aporta información con el fin de facilitar la realización de estas.

Este proyecto es una prueba de concepto en la cual se demuestra el potencial de esta tecnología y sus posibles aplicaciones en el mundo real, tanto en problemas industriales como cotidianos. Ofrece una herramienta de ayuda capaz de mostrar recursos multimodales (audio, video, texto...) obteniéndolos remotamente mediante instrucciones de voz, haciendo uso de un protocolo simple y conocido como es el protocolo SIP y desarrollada sobre una plataforma móvil ampliamente extendida, que permite el uso de esta aplicación en distintos tipos de dispositivos.

6.2. Trabajo futuro

A partir de esta base las posibilidades de desarrollo son tan amplias como los problemas que surjan en cualquiera de los campos donde se pueda aplicar esta clase de tecnologías.

El próximo paso de este proyecto será insertar en la plataforma N-Terac un nuevo módulo para plasmar objetos 3D de realidad aumentada en el stream de video del service node. Para ello, este módulo se servirá del stream de video que proporciona el media client y deberá ser capaz de detectar ciertos patrones, que usará para hacer el tracking, o seguimiento y plasmará sobre este el objeto 3D que sea adecuado.

Glosario de acrónimos

- **ASR**: Automatic Scpeech Recognition
- **TTS**: Text To Speech
- **QoS**: Quality of Service
- **SIP**: Session Initiation Protocol
- **HTTP**: Hypertext Transfer Protocol
- **URL**: Uniform Resource Locator
- **BBDD**: Data Base
- **NAS**: Network-Attached Storage
- **IU**: Interfaz de Usuario

Bibliografía

[1] James F Kurose, Keith W Ross, Carolina Mañoso Hierro, Ángel Pérez de Madrid y Pablo, and Luis Marrone. *Redes de computadoras: un enfoque descendente*. Addison Wesley, 2010.